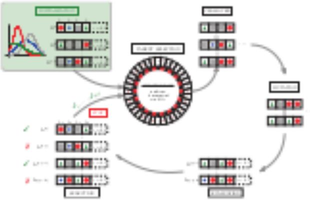
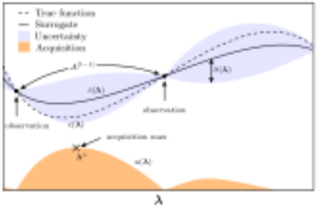
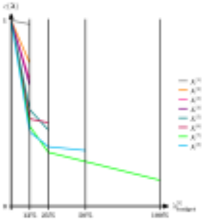
HPO – MANY APPROACHES

- Evolutionary algorithms
- Bayesian / model-based optimization
- Multi-fidelity optimization, e.g. Hyperband





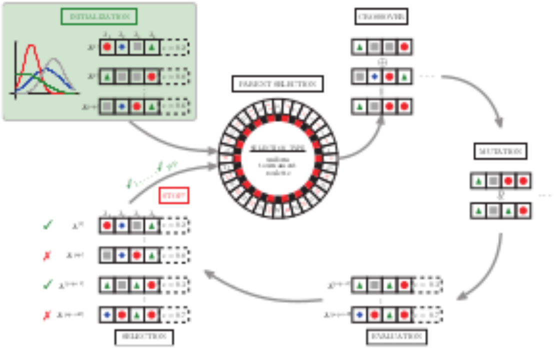


HPO methods can be characterized by:

- how the exploration vs. exploitation trade-off is handled
- how the inference vs. search trade-off is handled

Further aspects: Parallelizability, local vs. global behavior, handling of noisy observations, multifidelity and search space complexity.

EVOLUTIONARY STRATEGIES

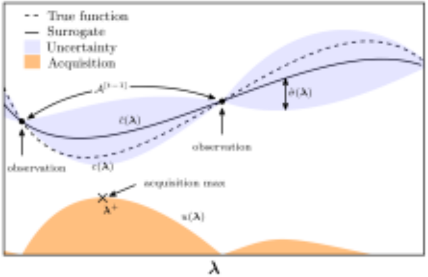


- Are a class of stochastic population-based optimization methods inspired by the concepts of biological evolution
- Are applicable to HPO since they do not require gradients
- Mutation is the (randomized) change of one or a few HP values in a configuration.
- Crossover creates a new HPC by (randomly) mixing the values of two other configurations.

BAYESIAN OPTIMIZATION

BO sequentially iterates:

1. **Approximate $\lambda \mapsto c(\lambda)$** by (nonlin) regression model $\hat{c}(\lambda)$, from evaluated configurations (archive)
2. **Propose candidates** via optimizing an acquisition function that is based on the surrogate $\hat{c}(\lambda)$
3. **Evaluate** candidate(s) proposed in 2, then go to 1

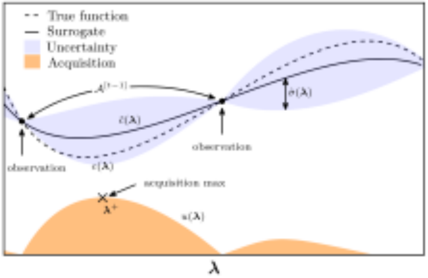


Important trade-off: **Exploration** (evaluate candidates in under-explored areas) vs. **exploitation** (search near promising areas)

BAYESIAN OPTIMIZATION

Surrogate Model:

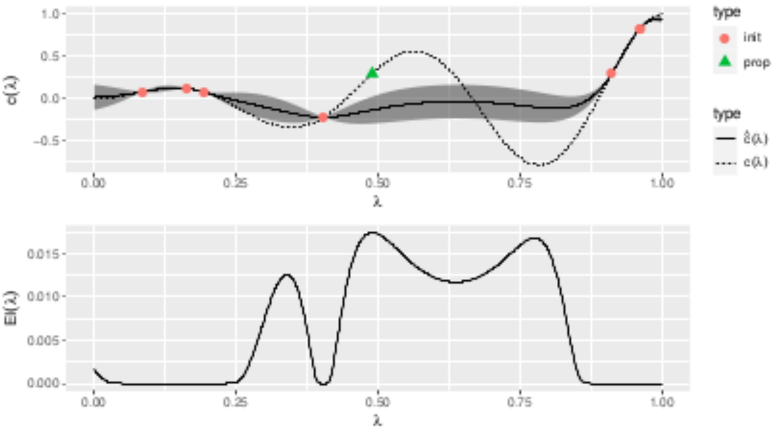
- Probabilistic modeling of $C(\boldsymbol{\lambda}) \sim (\hat{c}(\boldsymbol{\lambda}), \hat{\sigma}(\boldsymbol{\lambda}))$ with posterior mean $\hat{c}(\boldsymbol{\lambda})$ and uncertainty $\hat{\sigma}(\boldsymbol{\lambda})$.
- Typical choices for numeric spaces are Gaussian Processes; random forests for mixed spaces



Acquisition Function:

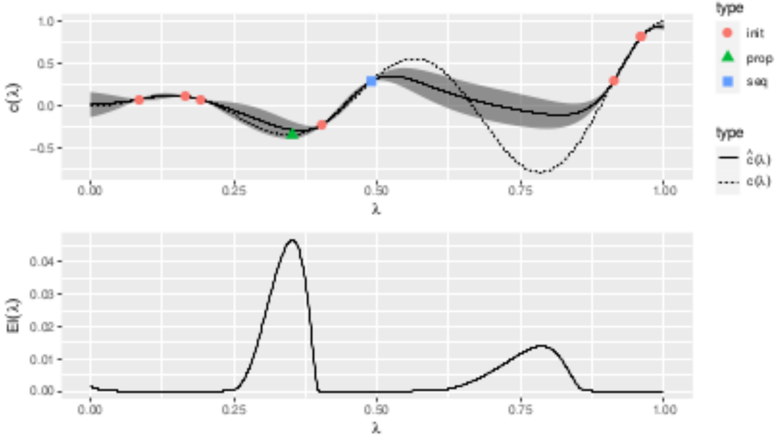
- Balance exploration (high $\hat{\sigma}$) vs. exploitation (low $\hat{c}$).
- Lower confidence bound (LCB): $a(\boldsymbol{\lambda}) = \hat{c}(\boldsymbol{\lambda}) - \kappa \cdot \hat{\sigma}(\boldsymbol{\lambda})$
- Expected improvement (EI): $a(\boldsymbol{\lambda}) = \mathbb{E}\left[\max\left\{c_{\min} - C(\boldsymbol{\lambda}), 0\right\}\right]$ where ($c_{\min}$ is best cost value from archive)
- Optimizing $a(\boldsymbol{\lambda})$ is still difficult, but cheap(er)

BAYESIAN OPTIMIZATION



Upper plot: The surrogate model (black, solid) models the *unknown* relationship between input and output (black, dashed) based on the initial design (red points).

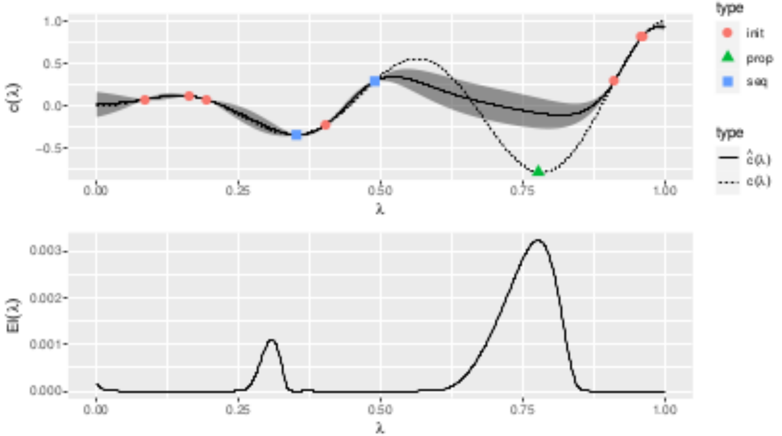
Lower plot: Mean and variance of the surrogate model are used to derive the expected improvement (EI) criterion. The point that maximizes the EI is proposed (green point).



Upper plot: The surrogate model (black, solid) models the *unknown* relationship between input and output (black, dashed) based on the initial design (red points).

Lower plot: Mean and variance of the surrogate model are used to derive the expected improvement (EI) criterion. The point that maximizes the EI is proposed (green point).

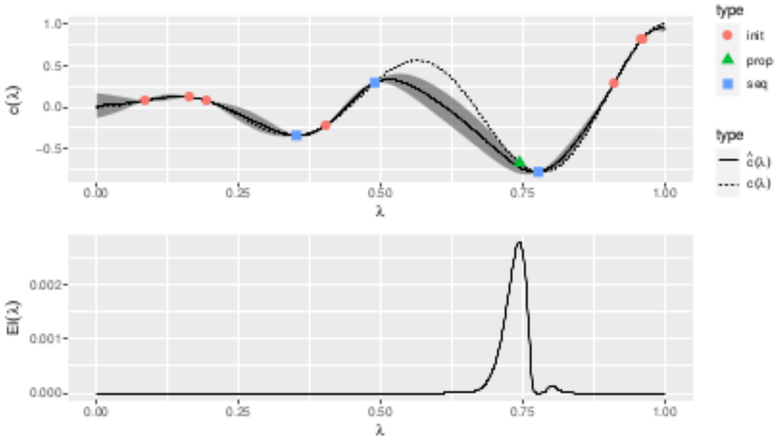
BAYESIAN OPTIMIZATION



Upper plot: The surrogate model (black, solid) models the *unknown* relationship between input and output (black, dashed) based on the initial design (red points).

Lower plot: Mean and variance of the surrogate model are used to derive the expected improvement (EI) criterion. The point that maximizes the EI is proposed (green point).

BAYESIAN OPTIMIZATION



Upper plot: The surrogate model (black, solid) models the *unknown* relationship between input and output (black, dashed) based on the initial design (red points).

Lower plot: Mean and variance of the surrogate model are used to derive the expected improvement (EI) criterion. The point that maximizes the EI is proposed (green point).

BAYESIAN OPTIMIZATION / 2

Since we use the sequentially updated surrogate model predictions of performance to propose new configurations, we are guided to “interesting” regions of Λ and avoid irrelevant evaluations:

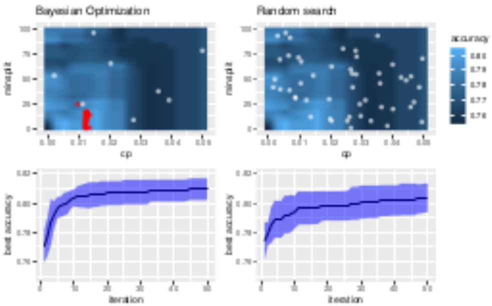


Figure: Tuning complexity and minimal node size for splits for CART on the `titanic` data (10-fold CV maximizing accuracy).
Left panel: BO, 50 configurations; right panel: random search, 50 iterations.
Top panel: one run (initial design of BO is white); bottom panel: mean $\pm$ std of 10 runs.

BAYESIAN OPTIMIZATION / 3

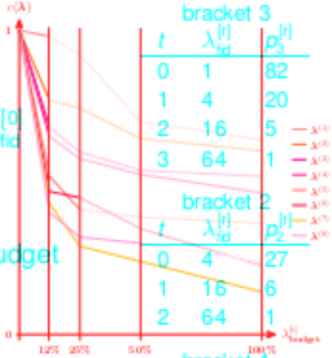
- Prerequiste: Fidelity HP λ_{fid}, i.e., a component of λ, which influences the computational cost of the fitting procedure in a monotonically increasing manner
- Methods of multifidelity optimization in HPO are all tuning approaches that can efficiently handle a $\mathcal{I}$ with a HP λ_{fid}
- The lower we set λ_{fid}, the more points we can explore in our search space, albeit with much less reliable information w.r.t. their true performance.
- We assume to know box-constraints of λ_{fid}, so $\lambda_{\text{fid}} \in [\lambda_{\text{fid}}^{\text{low}}, \lambda_{\text{fid}}^{\text{upp}}]$, where the upper limit implies the highest fidelity returning values closest to the true objective value at the highest computational cost.

MULTIFIDELITY OPTIMIZATION

- Prerequiste: Fidelity HP λ_{fid}, i.e., a component of $\boldsymbol{\lambda}$, which influences the computational cost of the fitting procedure in a monotonically increasing manner
- Methods of multifidelity optimization in HPO are all tuning approaches that can efficiently handle a $\mathcal{I}$ with a HP λ_{fid}
- The lower we set λ_{fid}, the more points we can explore in our search space, albeit with much less reliable information w.r.t. their true performance.
- We assume to know box-constraints of λ_{fid}, so $\lambda_{\text{fid}} \in [\lambda_{\text{fid}}^{\text{low}}, \lambda_{\text{fid}}^{\text{upp}}]$, where the upper limit implies the highest fidelity returning values closest to the true objective value at the highest computational cost.

SUCCESSIVE HALVING

- Races down set of HPCs to the best
- Idea: Discard bad configurations early
- Train HPCs with fraction of full budget (SGD epochs, training set size); the control param for this is called **multi-fidelity HP**
- Continue with better $1/\eta$ fraction of HPCs (w.r.t $\widehat{\mathrm{GE}}$); with η times budget (usually $\eta = 2, 3$)
- Repeat until budget depleted or single HPC remains



HYPERPARAMETER OPTIMIZATION – HYPERBAND

Problem with SH

- Good HPCs could be killed off too early, depends on evaluation schedule

Solution: Hyperband

- Repeat SH with different start budgets $\lambda_{fid}^{[0]}$ and initial number of HPCs $p^{[0]}$
- Each SH run is called bracket
- Each bracket consumes ca. the same budget

For $\eta = 4$

bracket 3

t	$\lambda_{fid}^{[t]}$	$p_3^{[t]}$
0	1	82
1	4	20
2	16	5
3	64	1

bracket 2

t	$\lambda_{fid}^{[t]}$	$p_2^{[t]}$
0	4	27
1	16	6
2	64	1

bracket 1

t	$\lambda_{fid}^{[t]}$	$p_1^{[t]}$
0	16	10
1	64	2

bracket 0

t	$\lambda_{fid}^{[t]}$	$p_0^{[t]}$
0	64	5

MULTIFIDELITY OPTIMIZATION – HYPERBAND

Problem with SH

- Good HPCs could be killed off too early, depends on evaluation schedule

Solution: Hyperband

- Repeat SH with different start budgets $\lambda_{\text{fid}}^{[0]}$ and initial number of HPCs $p^{[0]}$
- Each SH run is called bracket
- Each bracket consumes ca. the same budget

For $\eta = 4$

bracket 3

t	$\lambda_{\text{fid}}^{[t]}$	$p_3^{[t]}$
0	1	82
1	4	20
2	16	5
3	64	1

bracket 2

t	$\lambda_{\text{fid}}^{[t]}$	$p_2^{[t]}$
0	4	27
1	16	6
2	64	1

bracket 1

t	$\lambda_{\text{fid}}^{[t]}$	$p_1^{[t]}$
0	16	10
1	64	2

bracket 0

t	$\lambda_{\text{fid}}^{[t]}$	$p_0^{[t]}$
0	64	5

... techniques besides model-based optimization and the hyperband algorithm are:

- ... simulated annealing
- ... ms / CMAES
- Iterated F-Racing
- Many more . . .

For more information see *Hyperparameter Optimization: Foundations, Algorithms, Best Practices and Open Challenges*, Bischl (2021)